至此，便将一个实际的逻辑问题抽象成一个逻辑函数了。而且，这个逻辑函数首先是以真值表的形式给出的。

二、写出逻辑函数式

为便于对逻辑函数进行化简和变换，需要把真值表转换为对应的逻辑函数式。转换的方法已在第二章中讲过。

三、选定器件的类型

为了产生所需要的逻辑函数，既可以用小规模集成的门电路组成相应的逻辑电路，也可以用中规模集成的常用组合逻辑器件或可编程逻辑器件等构成相应的逻辑电路。应该根据对电路的具体要求和器件的资源情况决定采用哪一种类型的器件。

四、将逻辑函数化简或变换成适当的形式

在使用小规模集成的门电路进行设计时，为获得最简单的设计结果，应将函数式化成最简形式，即函数式中相加的乘积项最少，而且每个乘积项中的因子也最少。如果对所用器件的种类有附加的限制（例如只允许用单一类型的与非门），则还应将函数式变换成与器件种类相适应的形式（例如将函数式化作与非－与非形式）。

在使用中规模集成的常用组合逻辑电路设计电路时，需要将函数式变换为适当的形式，以便能用最少的器件和最简单的连线接成所要求的逻辑电路。在下一节中将会看到，每一种中规模集成器件的逻辑功能都可以写成一个逻辑函数式。在使用这些器件设计组合逻辑电路时，应该将待产生的逻辑函数变换成与所用器件的逻辑函数式相同或类似的形式。具体做法将在下一节中介绍。

有关使用存储器和可编程逻辑器件设计组合逻辑电路的具体做法，在后面的章节中再做介绍。

目前用于逻辑设计的计算机辅助设计软件几乎都具有对逻辑函数进行化简或变换的功能，因而，在采用计算和辅助设计时，逻辑函数的化简和变换都是由计算机自动完成的。

五、根据化简或变换后的逻辑函数式，画出逻辑电路的连接图

至此，原理性设计（或称逻辑设计）已经完成。

六、工艺设计

为了将逻辑电路实现为具体的电路装置，还需要做一系列的工艺设计工作，包括设计印刷电路板、机箱、面板、电源、显示电路、控制开关等。最后还必须完成组装、调试。这部分内容请读者自行参阅有关资料，这里就不做具体的介绍了。

图 4.2.2 中以方框图的形式总结了逻辑设计的过程。应当指出，上述的设计步骤并不是一成不变的。例如，有的设计要求直接以真值表的形式给出，就不用进行逻辑抽象了。又如，有的问题逻辑关系比较简单、直观，也可以不经过逻

辑真值表而直接写出函数式来。

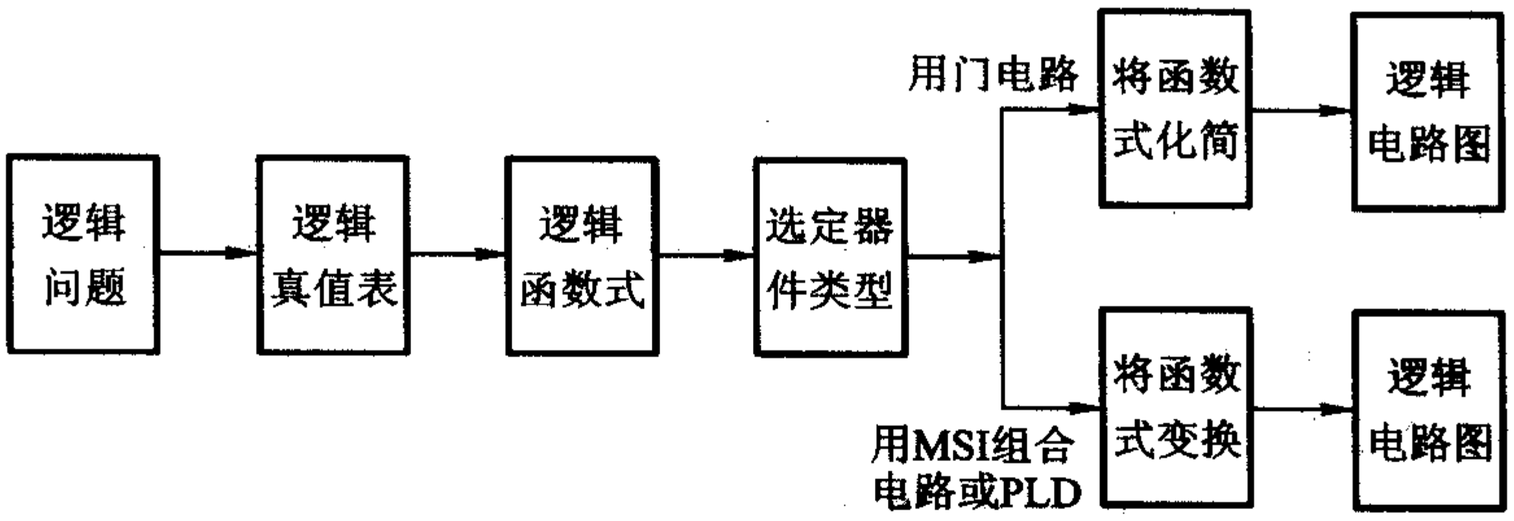

图 4.2.2 组合逻辑电路的设计过程

【例 4.2.2】 设计一个监视交通信号灯工作状态的逻辑电路。每一组信号灯均由红、黄、绿三盏灯组成，如图 4.2.3 所示。正常工作情况下，任何时刻必有一盏灯点亮，而且只允许有一盏灯点亮。而当出现其他五种点亮状态时，电路发生故障，这时要求发出故障信号，以提醒维护人员前去修理。

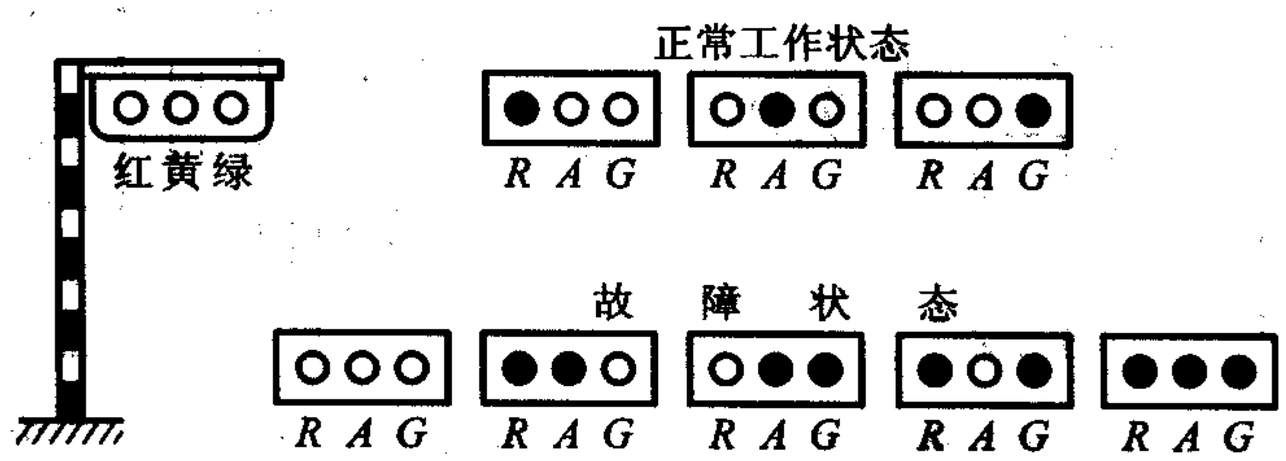

图 4.2.3 交通信号灯的正常工作状态与故障状态

解： (1) 首先进行逻辑抽象。

取红、黄、绿三盏灯的状态为输入变量，分别用 R、A、G 表示，并规定灯亮时为 **1**，不亮时为 **0**。取故障信号为输出变量，以 Z 表示之，并规定正常工作状态下 Z 为 **0**，发生故障时 Z 为 **1**。

根据题意可列出表 4.2.2 所示的逻辑真值表。

表 4.2.2 例 4.2.2 的逻辑真值表

R	A	G	Z
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0

续表

R	*A*	*G*	*Z*
1	0	1	1
1	1	0	1
1	1	1	1

(2) 写出逻辑函数式。

由表 4.2.2 知

$$Z = R'A'G' + R'AG + RA'G + RAG' + RAG \tag{4.2.2}$$

(3) 选定器件类型为小规模集成门电路。

(4) 将式(4.2.2)化简后得到

$$Z = R'A'G' + RA + RG + AG \tag{4.2.3}$$

(5) 根据式(4.2.3)的化简结果画出逻辑电路图,得到图 4.2.4 所示的电路。

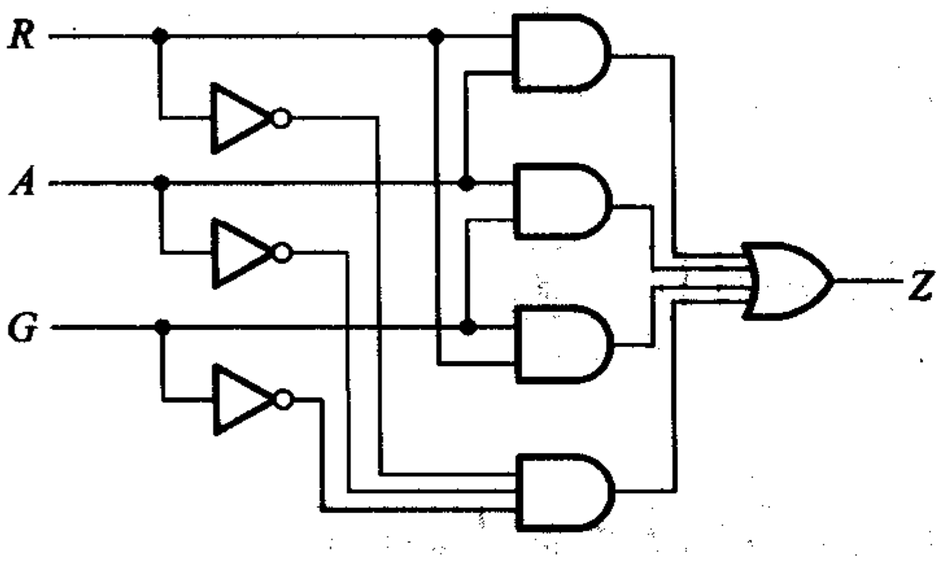

图 4.2.4　例 4.2.2 的逻辑图之一

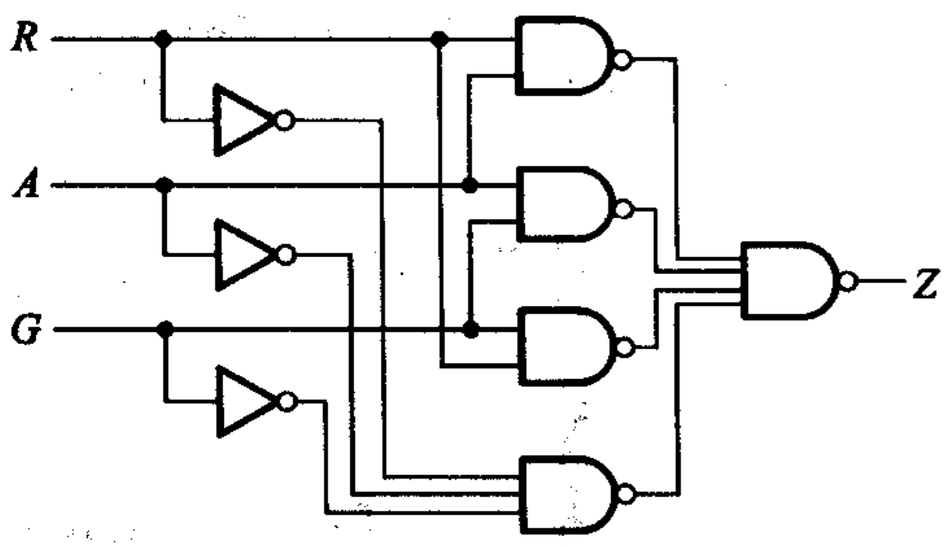

图 4.2.5　例 4.2.2 的逻辑图之二

由于式(4.2.3)为**最简与或**表达式,所以只有在使用**与**门和**或**门组成电路时才得到最简单的电路。如果要求用其他类型的门电路来组成这个逻辑电路,则为了得到最简单的电路,化简的结果亦需相应地改变。

例如,在要求全部用**与非**门组成这个逻辑电路时,就应当将函数式化为最简**与非 - 与非**表达式。这种形式通常可以通过将**与或**表达式两次求反得到。在上例中,将式(4.2.3)两次求反后得到

$$\begin{aligned} Z &= ((R'A'G' + RA + RG + AG)')' \\ &= ((R'A'G')'(RA)'(RG)'(AG)')' \end{aligned} \tag{4.2.4}$$

根据式(4.2.4)即可画出全部用**与非**门和反相器组成的逻辑电路,如图 4.2.5 所示。

如果要求用**与或非**门实现这个逻辑电路,那么就必须将式(4.2.3)化为最简**与或非**表达式。在第一章里我们曾经讲过,最简的**与或非**表达式可以通过合并卡诺图上的 **0**,然后求反而得到。为此,将函数 *Z* 的卡诺图画出,如图 4.2.6 所示。将图中的 **0** 合并、求反得到

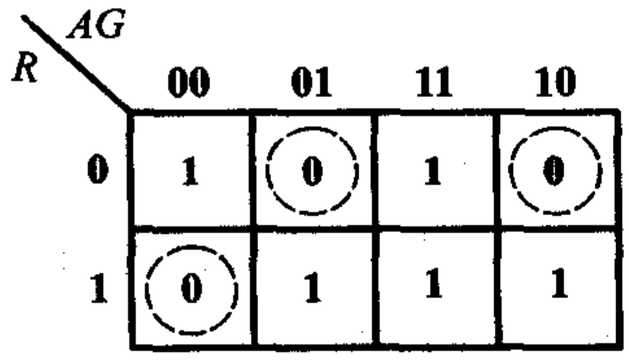

图 4.2.6 例 4.2.2 的卡诺图

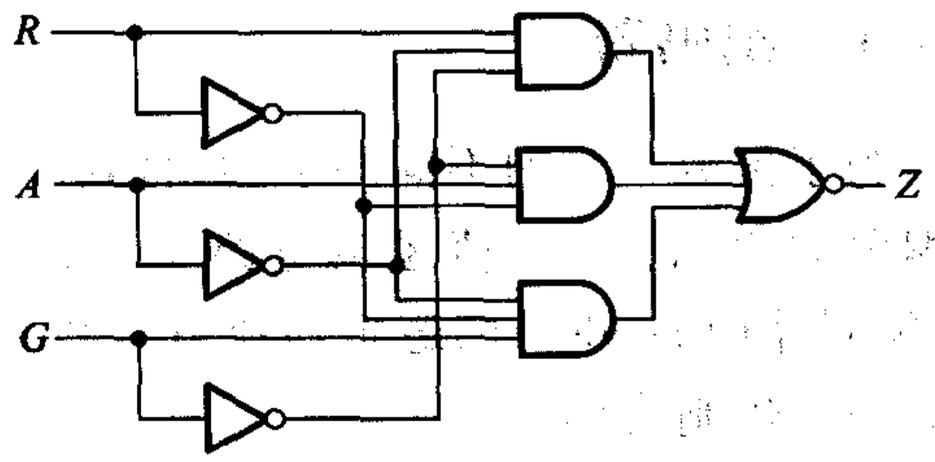

图 4.2.7 例 4.2.2 的逻辑图之三

$$Z = (RA'G' + R'AG' + R'A'G)' \tag{4.2.5}$$

按照式(4.2.5)画出的用与或非门组成的逻辑电路图如图 4.2.7 所示。

对于一些复杂的组合逻辑电路,往往已经难于用一组方程式完全描述它们的逻辑功能了,因而在设计这些逻辑电路时,通常采用“自顶向下”与“自底向上”相结合的设计方法。这两种方法都是首先将电路逐级分解为若干个简单的模块,然后再将这些模块设计好并连接起来。这些简单的模块电路都可以用这一节所讲的设计方法设计出来。有关“自顶向下”和“自底向上”的设计方法在后面还会做进一步的讲解。

复习思考题

R4.2.1 什么是“逻辑抽象”?它包含哪些内容?

R4.2.2 对于同一个实际的逻辑问题,两个同学经过逻辑抽象得到的逻辑函数不完全相同,这是为什么?

4.3 若干常用的组合逻辑电路

由于人们在实践中遇到的逻辑问题层出不穷,因而为解决这些逻辑问题而设计的逻辑电路也不胜枚举。然而我们发现,其中有些逻辑电路经常、大量地出现在各种数字系统当中。这些电路包括编码器、译码器、数据选择器、数值比较器、加法器、函数发生器、奇偶校验器/发生器等。为了使用方便,已经将这些逻辑电路制成了中、小规模集成的标准化集成电路产品。在设计大规模集成电路时,也经常调用这些模块电路已有的、经过使用验证的设计结果,作为所设计电路的组成部分。下面就分别介绍一下这些电路的工作原理和使用方法。

4.3.1　编码器

为了区分一系列不同的事物，将其中的每个事物用一个二值代码表示，这就是编码的含意。在二值逻辑电路中，信号都是以高、低电平的形式给出的。因此，编码器(Encoder)的逻辑功能就是将输入的每一个高、低电平信号编成一个对应的二进制代码。

一、普通编码器

目前经常使用的编码器有普通编码器和优先编码器两类。在普通编码器中，任何时刻只允许输入一个编码信号，否则输出将发生混乱。

现以3位二进制普通编码器为例，分析一下普通编码器的工作原理。图4.3.1是3位二进制编码器的框图，它的输入是$I_0 \sim I_7$八个高电平信号，输出是3位二进制代码$Y_2Y_1Y_0$。为此，又将它称为8线-3线编码器。输出与输入的对应关系由表4.3.1给出。

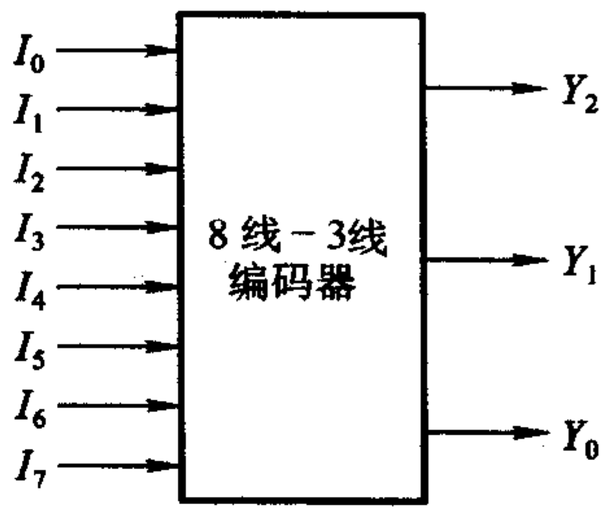

图4.3.1　3位二进制(8线-3线)编码器的框图

表4.3.1　3位二进制编码器的真值表

输入								输出		
I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	Y_2	Y_1	Y_0
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

将表4.3.1所示的真值表写成对应的逻辑式得到

$$\begin{cases} Y_2 = I_0'I_1'I_2'I_3'I_4I_5'I_6'I_7' + I_0'I_1'I_2'I_3'I_4'I_5I_6'I_7' \\ \quad + I_0'I_1'I_2'I_3'I_4'I_5'I_6I_7' + I_0'I_1'I_2'I_3'I_4'I_5'I_6'I_7 \\ Y_1 = I_0'I_1'I_2I_3'I_4'I_5'I_6'I_7' + I_0'I_1'I_2'I_3I_4'I_5'I_6'I_7' \\ \quad + I_0'I_1'I_2'I_3'I_4'I_5I_6'I_7' + I_0'I_1'I_2'I_3'I_4'I_5'I_6I_7' \\ Y_0 = I_0'I_1I_2'I_3'I_4'I_5'I_6'I_7' + I_0'I_1'I_2'I_3I_4'I_5'I_6'I_7' \\ \quad + I_0'I_1'I_2'I_3'I_4'I_5I_6'I_7' + I_0'I_1'I_2'I_3'I_4'I_5'I_6'I_7 \end{cases} \tag{4.3.1}$$

如果任何时刻 $I_0 \sim I_7$ 当中仅有一个取值为 1，即输入变量取值的组合仅有表 4.2.1 中列出的八种状态，则输入变量为其他取值下其值等于 1 的那些最小项均为约束项。利用这些约束项将式(4.3.1)化简，得到

$$\begin{cases} Y_2 = I_4 + I_5 + I_6 + I_7 \\ Y_1 = I_2 + I_3 + I_6 + I_7 \\ Y_0 = I_1 + I_3 + I_5 + I_7 \end{cases} \tag{4.3.2}$$

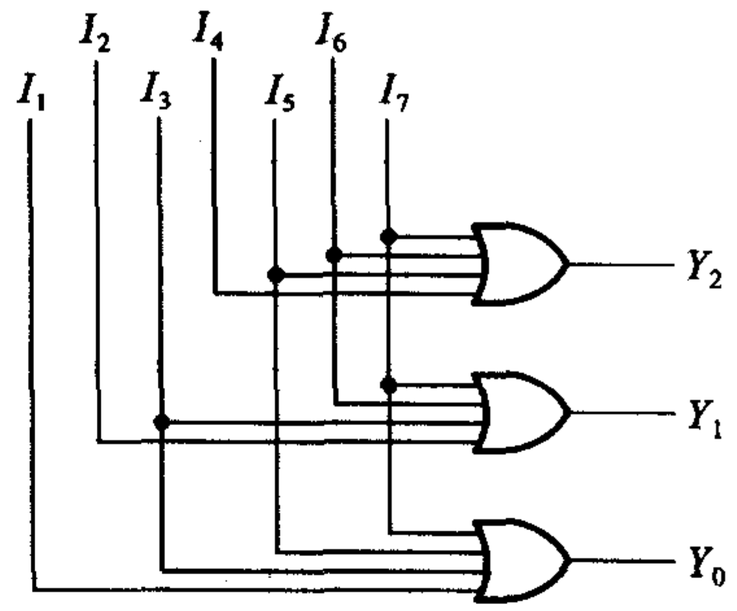

图 4.3.2 3 位二进制编码器

图 4.3.2 就是根据式(4.3.2)得出的编码器电路。这个电路是由三个**或**门组成的。

二、优先编码器

在优先编码器(priority encoder)电路中，允许同时输入两个以上的编码信号。不过在设计优先编码器时已经将所有的输入信号按优先顺序排了队，当几个输入信号同时出现时，只对其中优先权最高的一个进行编码。

图 4.3.3 给出了 8 线 -3 线优先编码器 74HC148 的逻辑图。如果不考虑由门 G_1、G_2 和 G_3 构成的附加控制电路，则编码器电路只有图中虚线框以内的这一部分。

由图 4.3.3 写出输出的逻辑式，即得到

$$\begin{cases} Y_2' = ((I_4 + I_5 + I_6 + I_7) \cdot S)' \\ Y_1' = ((I_2 I_4' I_5' + I_3 I_4' I_5' + I_6 + I_7) \cdot S)' \\ Y_0' = ((I_1 I_2' I_4' I_6' + I_3 I_4' I_6' + I_5 I_6' + I_7) \cdot S)' \end{cases} \tag{4.3.3}$$

为了扩展电路的功能和增加使用的灵活性，在 74HC148 的逻辑电路中附加了由门 G_1、G_2 和 G_3 组成的控制电路。其中 S'为选通输入端，只有在 $S' = 0$ 的条件下，编码器才能正常工作。而在 $S' = 1$ 时，所有的输出端均被封锁在高电平。

选通输出端 Y_S' 和扩展端 Y_{EX}'用于扩展编码功能。由图可知

$$Y_S' = (I_0' I_1' I_2' I_3' I_4' I_5' I_6' I_7' S)' \tag{4.3.4}$$

上式表明，只有当所有的编码输入端都是高电平(即没有编码输入)，而且 $S = 1$ 时，Y_S' 才是低电平。因此，Y_S' 的低电平输出信号表示“电路工作，但无编码输入”。

由图 4.3.3 还可以写出

$$\begin{aligned} Y_{EX}' &= ((I_0' I_1' I_2' I_3' I_4' I_5' I_6' I_7' S)' S)' \\ &= ((I_0 + I_1 + I_2 + I_3 + I_4 + I_5 + I_6 + I_7) \cdot S)' \end{aligned} \tag{4.3.5}$$

这说明只要任何一个编码输入端有低电平信号输入，且 $S = 1$，Y_{EX}'即为低电平。因此，Y_{EX}'的低电平输出信号表示“电路工作，而且有编码输入”。

根据式(4.3.3)、(4.3.4)和(4.3.5)可以列出表 4.3.2 所示的 74HC148 的

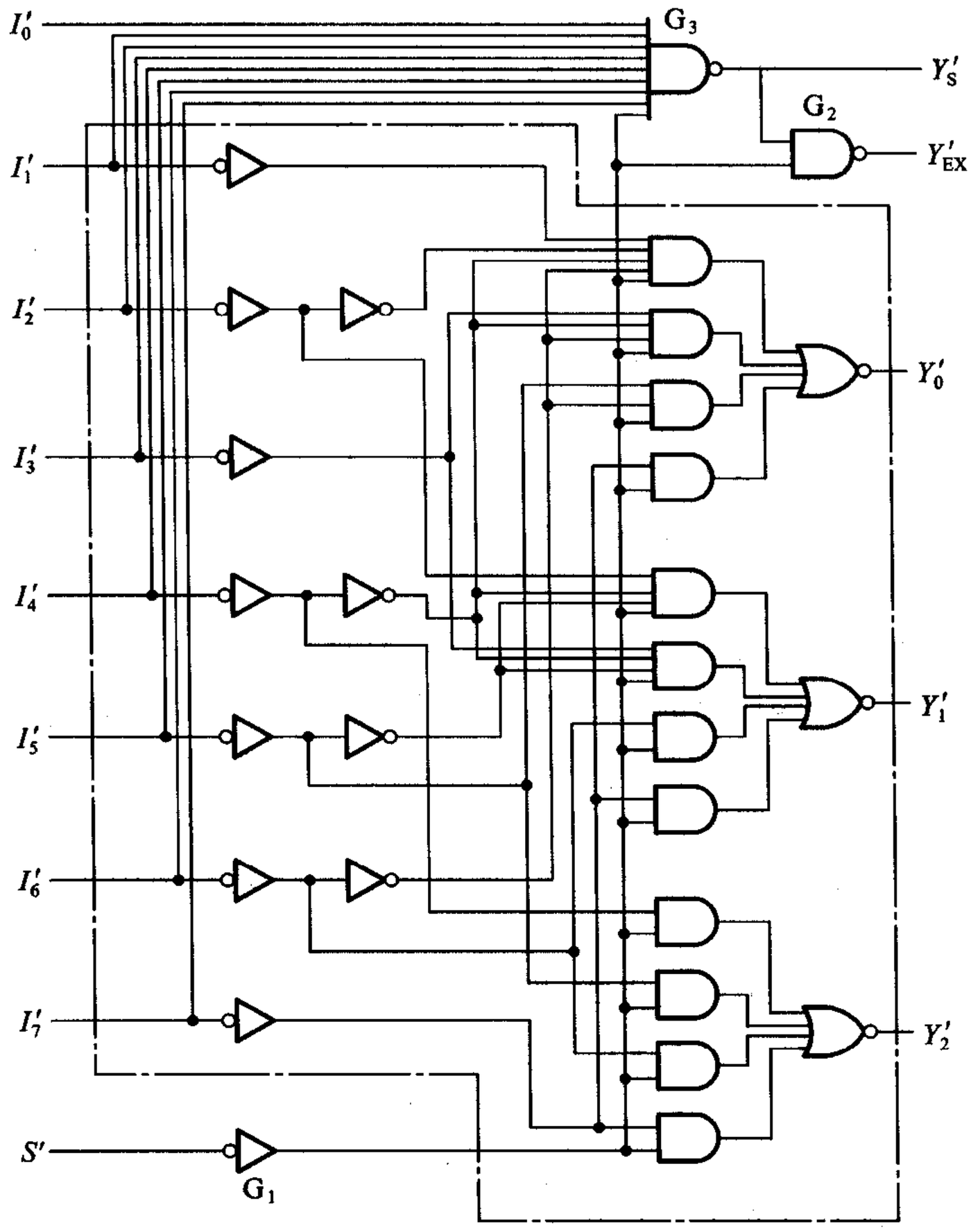

图 4.3.3　8 线 –3 线优先编码器 74HC148

功能表。它的输入和输出均以低电平作为有效信号。为了强调说明以低电平作为有效输入信号，有时也将反相器图形符号中表示反相的小圆圈画在输入端，如图 4.3.3 中左边一列反相器的画法。

表 4.3.2　74HC148 的功能表

输入									输出				
S'	I_0'	I_1'	I_2'	I_3'	I_4'	I_5'	I_6'	I_7'	Y_2'	Y_1'	Y_0'	Y_S'	Y_{EX}'
1	×	×	×	×	×	×	×	×	1	1	1	1	1
0	1	1	1	1	1	1	1	1	1	1	1	0	1
0	×	×	×	×	×	×	×	0	0	0	0	1	0
0	×	×	×	×	×	×	0	1	0	0	1	1	0
0	×	×	×	×	×	0	1	1	0	1	0	1	0
0	×	×	×	×	0	1	1	1	0	1	1	1	0

续表

输入									输出				
S'	I'_0	I'_1	I'_2	I'_3	I'_4	I'_5	I'_6	I'_7	Y'_2	Y'_1	Y'_0	Y'_S	Y'_{EX}
0	×	×	×	0	1	1	1	1	1	0	0	1	0
0	×	×	0	1	1	1	1	1	1	0	1	1	0
0	×	0	1	1	1	1	1	1	1	1	0	1	0
0	0	1	1	1	1	1	1	1	1	1	1	1	0

由表 4.3.2 中不难看出，在 $S'=0$ 电路正常工作状态下，允许 $I'_0 \sim I'_7$ 当中同时有几个输入端为低电平，即有编码输入信号。I'_7 的优先权最高，I'_0 的优先权最低。当 $I'_7=0$ 时，无论其他输入端有无输入信号（表中以 × 表示），输出端只给出 I'_7 的编码，即 $Y'_2Y'_1Y'_0=000$。当 $I'_7=1$、$I'_6=0$ 时，无论其余输入端有无输入信号，只对 I'_6 编码，输出为 $Y'_2Y'_1Y'_0=001$。其余的输入状态请读者自行分析。

表 4.3.2 中出现的三种 $Y'_2Y'_1Y'_0=111$ 的情况可以用 Y'_S 和 Y'_{EX} 的不同状态加以区分。

下面通过一个具体例子说明一下利用 Y'_S 和 Y'_{EX} 信号实现电路功能扩展的方法。

【例 4.3.1】 试用两片 74HC148 接成 16 线 -4 线优先编码器，将 $A'_0 \sim A'_{15}$ 16 个低电平输入信号编为 0000 ~ 1111 16 个 4 位二进制代码，其中 A'_{15} 的优先权最高，A'_0 的优先权最低。

解： 由于每片 74HC148 只有 8 个编码输入，所以需将 16 个输入信号分别接到两片上。现将 $A'_{15} \sim A'_8$ 8 个优先权高的输入信号接到第（1）片的 $I'_7 \sim I'_0$ 输入端，而将 $A'_7 \sim A'_0$ 8 个优先权低的输入信号接到第（2）片的 $I'_7 \sim I'_0$。

按照优先顺序的要求，只有 $I'_{15} \sim I'_8$ 均无输入信号时，才允许对 $I'_7 \sim I'_0$ 的输入信号编码。因此，只要将第（1）片的“无编码信号输入”信号 Y'_S 作为第（2）片的选通输入信号 S' 就行了。

此外，当第（1）片有编码信号输入时，它的 $Y'_{EX}=0$，无编码信号输入时 $Y'_{EX}=1$，正好可以用它作为输出编码的第四位，以区分 8 个高优先权输入信号和 8 个低优先权输入信号的编码。编码输出的低 3 位应为两片输出 Y'_2、Y'_1、Y'_0 的逻辑或。

依照上面的分析，便得到了图 4.3.4 所示的逻辑图。

由图 4.3.4 可见，当 $A'_{15} \sim A'_8$ 中任一输入端为低电平时，例如 $A'_{11}=0$，则片（1）的 $Y'_{EX}=0$，$Z_3=1$，$Y'_2Y'_1Y'_0=100$。同时片（1）的 $Y'_S=1$，将片（2）封锁，使它的输出 $Y'_2Y'_1Y'_0=111$。于是在最后的输出端得到 $Z_3Z_2Z_1Z_0=1011$。如果 $A'_{15} \sim A'_8$ 中同时有几个输入端为低电平，则只对其中优先权最高的一个信号编码。

当 $A'_{15} \sim A'_8$ 全部为高电平（没有编码输入信号）时，片（1）的 $Y'_S=0$，故片（2）

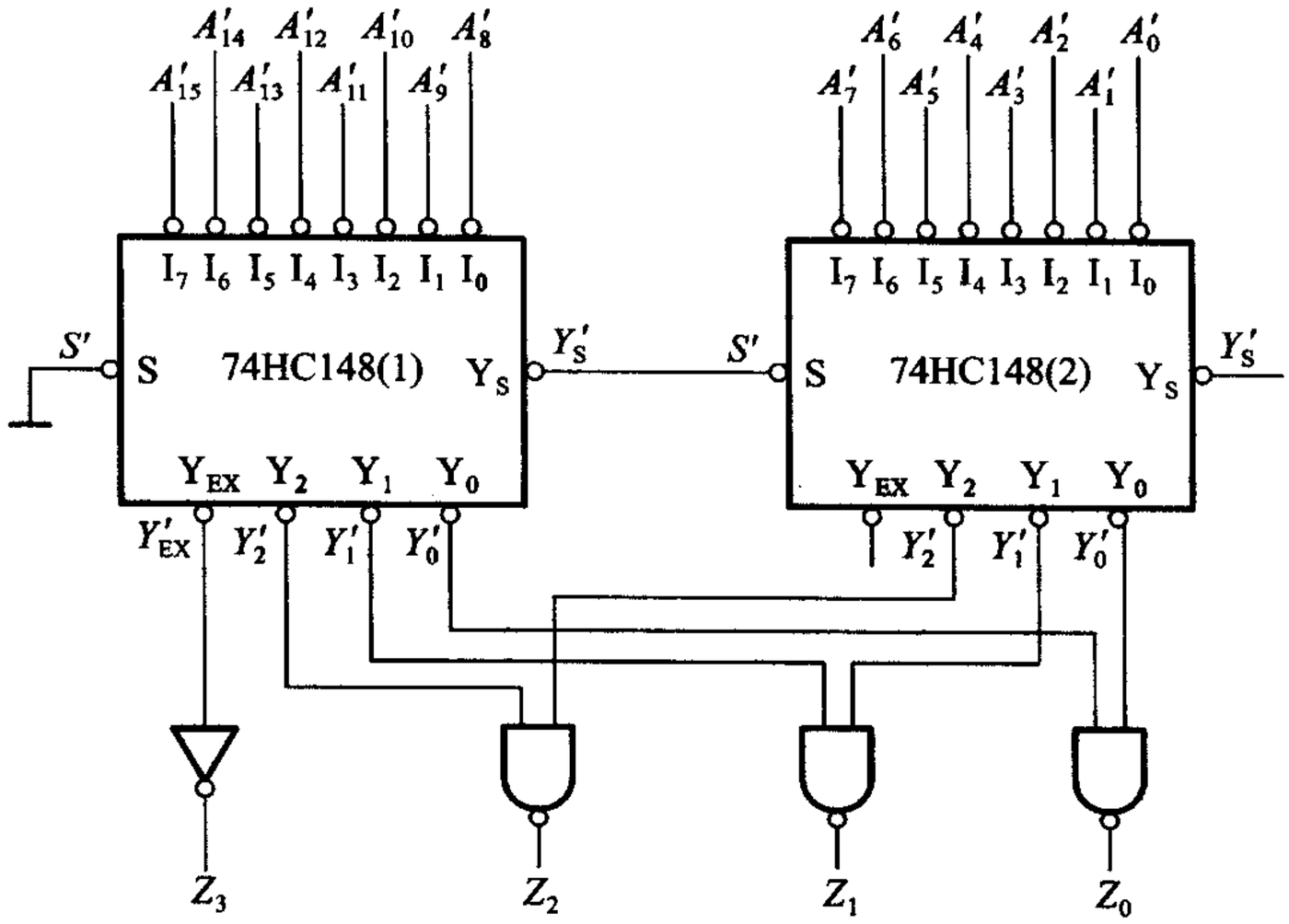

图 4.3.4　用两片 74HC148 接成的 16 线 -4 线优先编码器

的 $S'=\mathbf{0}$,处于编码工作状态,对 $A'_7 \sim A'_0$ 输入的低电平信号中优先权最高的一个进行编码。例如 $A'_5=\mathbf{0}$,则片(2)的 $Y'_2Y'_1Y'_0=\mathbf{010}$。而此时片(1)的 $Y'_{EX}=\mathbf{1}$,$Z_3=\mathbf{0}$。片(1)的 $Y'_2Y'_1Y'_0=\mathbf{111}$。于是在输出得到了 $Z_3Z_2Z_1Z_0=\mathbf{0101}$。

在常用的优先编码器电路中,除了二进制编码器以外,还有一类称为二 - 十进制优先编码器。它能将 $I'_0 \sim I'_9$10 个输入信号分别编成 10 个 BCD 代码。在 $I'_0 \sim I'_9$10个输入信号中 I'_9 的优先权最高,I'_0 的优先权最低。

在由中规模集成电路组成的应用电路中,习惯上采用逻辑框图来表示中规模集成电路器件,如图 4.3.4 中所示。在逻辑框图内部只标注输入、输出原变量的名称。如果以低电平作为有效的输入或输出信号,则于框图外部相应的输入或输出端处加画小圆圈,并在外部标注的输入或输出端信号名称上加非号“′”。

图 4.3.5 是二 - 十进制优先编码器 74LS147 的逻辑图。由图得到

$$\begin{cases} Y'_3=(I_8+I_9)' \\ Y'_2=(I_7I'_8I'_9+I_6I'_8I'_9+I_5I'_8I'_9+I_4I'_8I'_9)' \\ Y'_1=(I_7I'_8I'_9+I_6I'_8I'_9+I_3I'_4I'_5I'_8I'_9+I_2I'_4I'_5I'_8I'_9)' \\ Y'_0=(I_9+I_7I'_8I'_9+I_5I'_6I'_8I'_9+I_3I'_4I'_6I'_8I'_9+I_1I'_2I'_4I'_6I'_8I'_9)' \end{cases} \tag{4.3.6}$$

将式(4.3.6)化为真值表的形式,即得到表 4.3.3。由表可知,编码器的输出是反码形式的 BCD 码。优先权以 I'_9 为最高,I'_0 为最低。

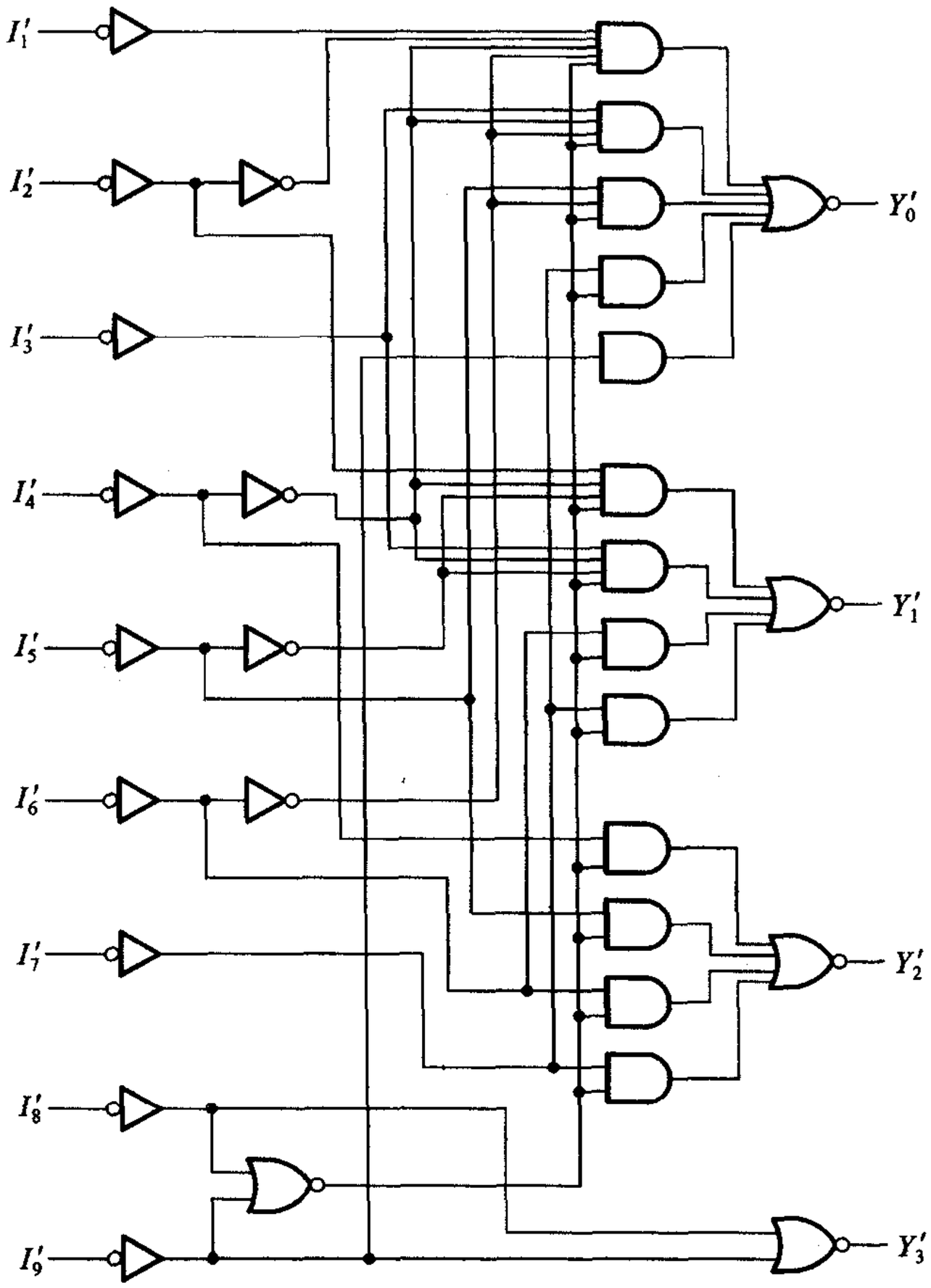

图 4.3.5 二－十进制优先编码器 74LS147

表 4.3.3 二－十进制编码器 74LS147 的功能表

输入									输出			
I_1'	I_2'	I_3'	I_4'	I_5'	I_6'	I_7'	I_8'	I_9'	Y_3'	Y_2'	Y_1'	Y_0'
1	1	1	1	1	1	1	1	1	1	1	1	1
×	×	×	×	×	×	×	×	0	0	1	1	0
×	×	×	×	×	×	×	0	1	0	1	1	1
×	×	×	×	×	×	0	1	1	1	0	0	0
×	×	×	×	×	0	1	1	1	1	0	0	1
×	×	×	×	0	1	1	1	1	1	0	1	0
×	×	×	0	1	1	1	1	1	1	0	1	1
×	×	0	1	1	1	1	1	1	1	1	0	0
×	0	1	1	1	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	1	0